The Role of Dominating Sets in Planning of Process Networks

“Abstract of Ph.D. Dissertation”

Zoltán Blázsik

University of Szeged
Szeged, 2008

1 Introduction

The analysis of network problems provided mathematically and practically significant results. The main subject of this thesis is an overview of the candidate's results in three topics. The first one is the Process Network Synthesis (PNS) problem, while the second one is an investigation of the existence of a perfect dominating set in de Bruijn graphs. The third topic of the thesis is the HPPIT problem.

2 The mathematical model of the PNS problem

Let M be a finite nonempty set, the set of materials. Let $O \subseteq \varphi'(M) \times \varphi'(M)$ be a nonempty set, the set of *operating units*, where $\varphi'(M)$ denotes the set of all nonempty subsets of M. For an $u = (\alpha, \beta) \in O$, α and β are called the *input-set* and *output-set* of u, respectively. The pair (M, O) is defined as *process graph* or P-*graph* in short. The vertex set of this directed bipartite graph is $M \cup O$, and the arc set is $A = A_1 \cup A_2$ with $A_1 = \{(X, Y) : Y = (\alpha, \beta) \in O \text{ and } X \in \alpha\}$ and $A_2 = \{(Y, X) : Y = (\alpha, \beta) \in O \text{ and } X \in \beta\}$. In case there exist vertices $X_1, X_2, ..., X_n$, such that $(X_1, X_2), (X_2, X_3), \ldots, (X_{n-1}, X_n)$ are arcs of process graph (M, O), then $[X_1, X_n]$ is defined to be a *path* from X_1 to X_n. Let the process graphs (m, o) and (M, O) be given; (m, o) is defined to be a *subgraph* of (M, O), if $m \subseteq M$, $o \subseteq O$, and $o \subseteq \varphi'(m) \times \varphi'(m)$. We can consider elements of O as abstract transformer machines which produce the set β from α.

Now we can define a *structural model* of PNS. Denote M' as the set of available materials, $P \subseteq M'$ is the set of *desired materials* and $R \subseteq M'$ is the set of *raw materials* ($P \cap R = \emptyset$). By the *structural model* of PNS, we mean the triplet, $\mathbf{M} = (P, R, O)$. Then, the process graph is (M, O), where $M = \cup\{\alpha \cup \beta : (\alpha, \beta) \in O\} \cup P \cup R$. (M, O) is the *P*-graph that represents the interconnections among the operating units of O. Furthermore, every feasible process network, producing the desired material set P from the given set R by using operating units from O, corresponds to a subgraph of (M, O). If additional constraints, e.g. the material balance, are disregarded, the subgraphs of (M, O), which can be assigned to the feasible processes, have common combinatorial properties. Such properties, explored in [18], are given below.

Subgraph (m, o) of (M, O) is called a *feasible solution* of $\mathbf{M} = (P, R, O)$ if the following properties are satisfied.

($\mathcal{A}1$) $P \subseteq m$,

($\mathcal{A}2$) $\forall X \in m$, $X \in R \Leftrightarrow$ there exists no (Y, X) arc in (m, o),

($\mathcal{A}3$) $\forall Y_0 \in o$, $\exists$ path $[Y_o, Y_n]$ with $Y_n \in P$,

($\mathcal{A}4$) $\forall X \in m$, $\exists(\alpha, \beta) \in o$ such that $X \in \alpha \cup \beta$.

Let us denote $S(\mathbf{M})$ the set of the feasible solutions of $\mathbf{M}$. It is easy to see that $S(\mathbf{M})$ is closed under the finite union. Consequently,

$$\cup\{(m, o) : (m, o) \in S(\mathbf{M})\}$$

is also a feasible solution provided that $S(\mathbf{M}) \neq \emptyset$; it is the greatest feasible solution with respect to the relation of subgraph ordering. This distinguished graph is called the *maximal structure* of $\mathbf{M}$.

The first PNS problem is the question, how we can obtain feasible solutions, and how we can generate the maximal structure from a structural model. In [14] and [19] a simple polynomial time algorithm is presented for the generation of the maximal structure, if it is exists.

The structural model of PNS problem $\mathbf{M} = (P, R, O)$ be given; moreover, let z be a positive real-valued weight function defined on S($\mathbf{M}$). The basic model is then

$$\min\{z((m, o)) : \ (m, o) \in S(\mathbf{M})\}. \tag{1}$$

The cost minimization of feasible solution of a process network is indeed essential. For this purpose, several papers appeared for solving PNS problems by global optimization methods (cf. [13] and [22]) and by combinatorial approaches based on the feasible graphs of processes (see, e.g., [14], [18], [21]). However, its solution is difficult in general.

Now, a class of PNS problems can be defined, when each operating unit has a positive fixed cost. We want to find a feasible solution with the minimum sum of costs of the units. Let $w : O \to R_+$ be a fixed weight function defined on O. Our objective function is

$$\min\{\sum_{u \in o} w(u) \ : \ (m, o) \in S(\mathbf{M})\}. \tag{2}$$

3 The PNS problem is NP–complete

It is a very important, fundamental question in theory of PNS, whether the combinatorial optimization problem (2) is in P. It is easy to check whether $(m, o) \in S(\mathbf{M})$, and what is the value of the objective function for (m, o), so (2) is in NP. We proved in [2] the fact, that (2) is NP–hard.

We referred to the problem (2) as a PNS_w-*problem*; we denote the class of such problems by PNS_w. In what follows, we define a subclass of PNS_w, which is equivalent to the class of the classical set covering problems.

Let us denote by PNS_{w1} the subclass of PNS_w for which a problem from PNS_w given by (P, R, O) and w is contained in PNS_{w1} if and only if $O \subseteq \wp'(R) \times \wp'(P)$. The meaning of this subclass can be given as follows:

It contains such process design problems in which the operating units use only the raw materials as inputs and yield only the desired materials as outputs; moreover, they perform in parallel. The following two statements were shown in [2]:

Theorem 3.1 *The class PNS_{w1} is equivalent to the class of the set covering problems.*

Obviously, if a PNS_{w1}-problem is equipped with the condition that each desired product can be produced by at most one operating unit in each solution-structure, then we can construct an equivalent set partitioning problem and vice versa. Since both the set covering and set partitioning problems are well-known to be NP–complete, the PNS_{w1}-problem must be NP-complete as well. This leads immediately to the next corollary.

Corollary 3.2 *The PNS problem is NP-hard.*

Let us observe that in a PNS_{w1}-problem the set of materials is divided into two disjoint sets P and R, and each operating unit has a nonempty subset of R as inputs and a nonempty subset of P as outputs. Generalizing this feature, we defined in [2] further subclasses of PNS_w-problems. Specifically, let $k \geq 1$ be an arbitrary fixed integer in order to consider the problems in which $M = M_1 \bigcup \cdots \bigcup M_{k+1}$ where the sets $M_1, \ldots, M_{k+1}$ are pairwise disjoint nonempty sets. Furthermore, let $O = O_1 \bigcup \cdots \bigcup O_k$ with $O_i \subseteq \wp'(M_1 \bigcup \cdots \bigcup M_i) \times \wp'(M_{i+1})$, $i = 1, \ldots, k$. We have called such a PNS_w-problem a PNS_{wk}-*problem.*

These PNS classes were the motivation for [20] and [27]. In these papers there is a proof to another direction; the authors have pointed out that (2) is a special case of the set covering problem, too. The (2) minsum version of weighted PNS problem is one of the NP–complete problems, thus we can investigate *well-solvable special classes* [28], [29], and *heuristic algorithms* [8].

4 Bounds for the number of the consistent decision-mappings

The first approach was to develop exponential time algorithms for solving the PNS problem. Exponential time algorithms based on the Branch and Bound technique were developed and studied for the PNS problem in [24] and [25], and the notion of the decision-mapping (see [15]) has been introduced. The P-graph (M, O) of $\mathbf{M}$ determines a function Δ of $M \setminus R$ into $\varphi'(O)$ as follows. For any material $X \in M \setminus R$, let

$$\Delta(X) = \{(\alpha, \beta) : (\alpha, \beta) \in O \ \& \ X \in \beta\}.$$

Let m be a subset of $M \setminus R$; furthermore, let $\delta(X)$ be a subset of $\Delta(X)$ for each $X \in m$. Mapping δ from set m into the set of subsets of O, $\delta[m] = \{(X, \delta(X)) : X \in m\}$, is called a *decision-mapping belonging* to $\mathbf{M}$; $\delta[m]$ is said to be *consistent* when $\delta(X) \cap \Delta(Y) \subseteq \delta(Y)$ is valid for all $X, Y \in m$, and the set of all consistent decision-mappings of $\mathbf{M}$ is denoted by $\Omega_{\mathbf{M}}$. In particular, if $\delta[m] \in \Omega_{\mathbf{M}}$ and $m = M \setminus R$, then sometimes we use the shorter notation δ instead of $\delta[M \setminus R]$. A decision-mapping can be visualized as a sequence of decisions, each of which is concerned with a single material involved in the process being synthesized; it identifies the set of operating units to be considered for producing directly the material of interest. The meaning of the consistency can be presented as follows. Material X is to be produced by operating units included in $\delta(X)$. Then, those operating units of $\delta(X)$ that also participate in the production of material Y, i.e., $\delta(X) \cap \Delta(Y)$, must be considered for the production of material Y, and thus, $\delta(Y) \supseteq \delta(X) \cap \Delta(Y)$.

We define the function *op* on $\Omega_{\mathbf{M}}$ for selecting the set of those operating units that are decided to produce any of the materials in set m based on the consistent decision-mapping $\delta[m]$. Formally, for any $\delta[m] \in \Omega_{\mathbf{M}}$,

$$op(\delta[m]) = \cup\{\delta(X) : X \in m\}.$$

Furthermore, we need the following functions. For any finite set of operating units o, let

$$mat^{in}(o) = \cup_{(\alpha,\beta)\in o}\alpha, \qquad mat^{out}(o) = \cup_{(\alpha,\beta)\in o}\beta.$$

Let $\delta_1[m_1]$ and $\delta_2[m_2]$ be arbitrary consistent decision-mappings. Then, $\delta_2[m_2]$ is called an *extension* of $\delta_1[m_1]$ if $m_1 \subseteq m_2$ and $\delta_1(X) = \delta_2(X)$ for all $X \in m_1$; this is denoted by $\delta_1[m_1] \leq \delta_2[m_2]$. In particular, if $\delta_1[m_1] \leq \delta_2[m_2]$ and $m_1 \subset m_2$, a *proper extension* exists; it is denoted by $\delta_1[m_1] < \delta_2[m_2]$. Relation extension is reflexive, antisymmetric and transitive; hence, it is a partial ordering on $\Omega_{\mathbf{M}}$. Let us denote the set of all maximal elements of this partially ordered set by $\Omega_{\mathbf{M}}^{\max}$.

Theorem 4.1 *For every $\emptyset \neq m \subseteq M \setminus R$, the number of the decision-mappings defined on m is $2^{\sum_{X\in m}|\Delta(X)|}$.*

Let us denote by $\tau(m)$ the number of the consistent decision-mappings defined on m.

Theorem 4.2 ([3]) *For every $\emptyset \neq m \subseteq M \setminus R$, $\tau(m) = 2^{|\cup\{\Delta(X):X\in m\}|}$.*

Remark 4.3 *In particular, if $m = M \setminus R$, then $\tau(m) = 2^{|O|}$. This shows that there is a strong relationship between the maximal consistent decision-mappings and the subsets of O. Indeed, it can be proved that mapping γ defined by $\gamma(\delta) = op(\delta)$ is a one-to-one mapping of $\Omega_{\mathbf{M}}^{\max}$ onto $\varphi(O)$ where $\varphi(O)$ denotes the set of all subsets of O.*

Regarding the relationship between the maximal decision-mappings and the feasible solutions, let us define mapping ρ in the following way. For any $(m, o) \in S(\mathbf{M})$, let $\rho(m, o) = \delta$ where δ is defined by

$$\delta(X) = \{u : u = (\alpha, \beta) \in o \;\&\; X \in \beta\}$$

for all $X \in M \setminus R$. It can be easily proved that ρ is a one-to-one mapping of $S(\mathbf{M})$ into $\Omega_{\mathbf{M}}^{\max}$. Therefore, $2^{|O|}$ is a trivial upper bound for $|S(\mathbf{M})|$.

Taking into account property (A2), this bound can be improved as follows. Let $(m, o) \in S(\mathbf{M})$ be an arbitrary feasible solution and $\rho(m, o) = \delta$. Then, (A2) implies the following inclusion:

$(\mathcal{A}'2)$ $$mat^{in}(op(\delta)) \subseteq mat^{out}(op(\delta)) \cup R.$$

Let us denote by $\tau'(m)$ the number of the consistent decision-mappings defined on m satisfying $(\mathcal{A}'2)$. Then $\tau'(m) \geq |S(\mathbf{M})|$.

Let $O = \{u_1, \ldots, u_n\}$, $M = \{X_1, \ldots, X_k\}$, $O(X_j) = \{u : u = (\alpha, \beta) \in O \;\&\; X_j \in \alpha\}$, for all $X_j \in M$, and for all $j \in \{1, \ldots, k\}$

$$A_j = \{\delta : \delta \in \Omega_{\mathbf{M}}^{\max} \;\&\; X_j \in mat^{in}(op(\delta)) \setminus (mat^{out}(op(\delta)) \cup R)\}.$$

Then, $(\mathcal{A}'2)$ is not satisfied by δ and the reason for it is that $X_j \in mat^{in}(op(\delta))$ and $X_j \notin mat^{out}(op(\delta)) \cup R$. For every $\emptyset \neq I \subseteq \{1, \ldots, k\}$, let us define the set A_I by $A_I = \cap_{i\in I} A_i$, and in particular, let $A_\emptyset = \Omega_{\mathbf{M}}^{\max}$. If $I = \{i_1, \ldots, i_l\}$, then

$$A_I = \left\{\delta : \delta \in \Omega_{\mathbf{M}}^{\max} \ \& \ \{X_{i_1}, \ldots, X_{i_l}\} \subseteq mat^{in}(op(\delta)) \setminus (mat^{out}(op(\delta)) \cup R)\right\}$$

Now we can count $\tau'(m)$ by the Inclusion-Exclusion Formula as an upper bound to $|S(\mathbf{M})|$.

Theorem 4.4 [3] $\tau'(m) = |\Omega_{\mathbf{M}}^{\max} \setminus (A_1 \cup A_2 \cup ... \cup A_k)| =$
$= \Sigma_{I\subseteq\{1,...k\}}(-1)^{|I|} \cdot |A_I|$.

Remark 4.5 *It is worth noting that the bound presented above is independent of the set of the required products. It is valid for arbitrary $P \subseteq M \setminus R$.*

Unfortunately, to count $|A_I|$ is a difficult problem. In the general case we have to cover $\{X_{i_1}, \ldots, X_{i_l}\}$ with such a system, $\alpha_{j_1}, \ldots, \alpha_{j_s}$, for which there are operating units $(\alpha_{j_t}, \beta_{j_t}) \in O$, $t = 1, \ldots, s$, with $\{X_{i_1}, \ldots, X_{i_l}\} \cap \beta_{j_t} = \emptyset$, $t = 1, \ldots, s$, and $|A_I|$ is equal to the number of such covering systems.

5 Explicit bounds for the number of feasible solutions of special PNS problem classes

The determination of $|A_I|$ is easier if we restrict ourselves to special classes of PNS problems. An interesting special case is the class containing separator type operating units, i.e., $|\alpha| = 1$ is valid for all $u = (\alpha, \beta) \in O$. Let us consider the set $I = \{i_1, \ldots, i_l\}$ again. Let $O^*(X_{i_j}) = O(X_{i_j}) \setminus (\cup_{i\in I} \Delta(X_i))$. Then, $O^*(X_{i_j})$ is the set of operating units such that they do not produce any material from $\{X_t : t \in I\}$ and each of them has X_{i_j} as input material.

Theorem 5.1 [3] *For separator type operating units*

$$|A_I| = \left(\prod_{t=1}^{l} \left(2^{|O^*(X_{i_t})|} - 1\right)\right) \cdot 2^{|O \setminus (\cup_{i\in I}\Delta(X_i)) \setminus (\cup_{i\in I} O(X_i))|}.$$

A model for the PNS problem with separator type operating units is called *Line model*, if

$$u_1 = (\alpha_1, \beta_1) \text{ with } \alpha_1 = X_1 \text{ and } \beta_1 = X_2,$$

$$u_k = (\alpha_k, \beta_k) \text{ with } \alpha_k = X_k \text{ and } \beta_k = X_{k-1},$$

and in general:

$$u_i = (\alpha_i, \beta_i) \text{ with } \alpha_i = X_i \text{ and } \beta_i = \{X_{i-1}, X_{i+1}\}, (2 \le i \le k-1).$$

In our second model we modify β_1 and β_k such that:

$$\beta_1 = \{X_2, X_k\} \text{ and } \beta_k = \{X_{k-1}, X_1\}.$$

Then we obtain a more symmetric another model called the *Chain model.*

Theorem 5.2 [4] *In the Line model* $|S(\mathbf{M})| \le L^{(1)}$*,*

where

$$L^{(1)} = 2^k + \sum_{1\le j\le \frac{k+1}{2}} (-1)^j \cdot \left[\sum_{\substack{0\le r\le j-1\\ k-3j+r+2\ge 0}} \binom{j-1}{r} \cdot \binom{k-2j}{j-r-2} \cdot 2^{k-3j+r+2} + \right.$$

$$+ \sum_{\substack{0\le r\le j-1\\ k-3j+r+1\ge 0}} \binom{j-1}{r} \cdot \binom{k-2j}{j-r-1} \cdot 2^{k-3j+r+1} +$$

$$\left. + \sum_{\substack{0\le r\le j-1\\ k-3j+r\ge 0}} \binom{j-1}{r} \cdot \binom{k-2j}{j-r} \cdot 2^{k-3j+r} \right]$$

$$= 1 + \sum_{2\le t\le k} \sum_{1\le q\le \min\{\frac{t}{2};k-t+1\}} \binom{t-q-1}{q-1} \cdot \binom{k-t+1}{q}.$$

Theorem 5.3 [4] *In the Chain model* $|S(\mathbf{M})| \le C^{(1)}$*,*

where

$$C^{(1)} = 2^k + \sum_{1\le j< \frac{k}{2}} (-1)^j \cdot \sum_{\substack{0\le r\le j-1\\ k-3j+r\ge 0}} \frac{k}{j} \cdot \binom{j}{r} \cdot \binom{k-2j-1}{j-r-1} \cdot 2^{k-3j+r} + e_k =$$

$$= 1 + \sum_{2\le t\le k} \left[\sum_{1\le q\le \min\{\frac{t}{2};k-t\}} \binom{t-q-1}{q-1} \cdot \binom{k-t-1}{q-1} + \right.$$

$$+\sum_{2\le i\le t}\ \sum_{1\le q\le \frac{t-i}{2}+1}\binom{t-i-q+1}{q-1}\cdot\binom{k-t-1}{q-1}+$$

$$\left.+\sum_{1\le i\le k-t}\ \sum_{1\le q\le \min\{\frac{t}{2};k-t-i+1\}}\binom{t-q-1}{q-1}\cdot\binom{k-t-i}{q-1}\right]+1,$$

where

$$e_k=\begin{cases}(-1)^{\frac{k}{2}}\cdot 2 & \text{, if } k \text{ is even,}\\ 0 & \text{, if } k \text{ is odd.}\end{cases}$$

In the general case, the Inclusion-Exclusion-Formula cannot be used because of complicate combinatorial and graph theory problems. In particular, the special structures of the considered classes allow also to count directly the required upper bounds.

$$L^{(2)}=1+\sum_{2\le t\le k}\ \sum_{1\le q\le \min\{\frac{t}{2};k-t+1\}}\binom{t-q-1}{q-1}\cdot\binom{k-t+1}{q}.$$

$$C^{(2)}=1+\sum_{2\le t\le k}\left[\sum_{1\le q\le \min\{\frac{t}{2};k-t\}}\binom{t-q-1}{q-1}\cdot\binom{k-t-1}{q-1}+\right.$$

$$+\sum_{2\le i\le t}\ \sum_{1\le q\le \frac{t-i}{2}+1}\binom{t-i-q+1}{q-1}\cdot\binom{k-t-1}{q-1}+$$

$$\left.+\sum_{1\le i\le k-t}\ \sum_{1\le q\le \min\{\frac{t}{2};k-t-i+1\}}\binom{t-q-1}{q-1}\cdot\binom{k-t-i}{q-1}\right]+1\,.$$

Summarizing the results, we obtained two nice combinatorial identity, $L^{(1)}=L^{(2)}$ and $C^{(1)}=C^{(2)}$.

6 Bottleneck and k-sum PNS-problems

In [6], we have shown that the k-sum version of the PNS problem is well-solvable, and thus, the PNS problem is such a particular case of the Min-sum problem which is NP-complete while its k-sum version is well-solvable for fixed k.

Let a reduced structural model of PNS problem $\mathbf{M}=(P,R,O)$ be given.

6.1 Bottleneck PNS-problem

We are to find such a feasible solution in which the weightest operating unit has the least weight. Formally, we want to solve

$$\min\{\ \max\{w(u) : u \in o\} : (m, o) \in S(\mathbf{M})\}. \tag{3}$$

Let $O = \{u_1, \ldots, u_n\}$. Without loss of generality, it can be supposed that $w(u_1) \leq w(u_2) \leq \cdots \leq w(u_n)$. For every positive integer $i (\leq n)$, let $O_i = \{u_1, \ldots, u_i\}$ and $M_i = mat(O_i)$. Furthermore, let $\mathbf{M}_i = (P, R, O_i)$.

Lemma 6.1 *If for some integer $1 \leq i \leq n$, $S(\mathbf{M}_i)$ has the maximal structure and $S(\mathbf{M}_{i-1})$ does not have the maximal structure, then u_i is included in every feasible solution in $S(\mathbf{M}_i)$.*

By the above statement, we could show [6] that the optimal value of (3) is $w(u_i)$.

Now, we can solve (3) by the following procedure.

Procedure 1.

- Initialization

  Let $i = 1$ and $O_i = \{u_1\}$.

- Iteration (i-th).

  Let $M_i = mat(O_i)$. Let us perform the Algorithm for Maximal Structure Generation for the structural model $\mathbf{M}_i = (P, R, O_i)$. If we have the maximal structure, then terminate; the maximal structure is an optimal solution and the optimal value is $w(u_i)$. In the opposite case, let $O_{i+1} = \{u_1, \ldots, u_{i+1}\}$, $i := i + 1$, and proceed to the next iteration.

Procedure 1. provides a better time complexity, $q(n) \cdot \log(n)$, where $q(n)$ denotes the time complexity of the Algorithm for Maximal Structure Generation.

6.2 The k-sum version of the PNS problem

Let k be a fixed positive integer. We are to find such a feasible solution for which the sum of weights of the k heaviest operating units is minimal.

To formalize the considered problem, let us denote by $u_{i_1}, \ldots, u_{i_k}$ the k heaviest operating units of (m, o). The *k-sum* PNS *problem* is then

$$\sum_{t=1}^{k} w(u_{i_t}) : (m, o) \in S(\mathbf{M}). \tag{4}$$

For solving (4), after a similar ordering of operating units let us fix such a linear ordering, denoted by $\preceq$, on the subsets of at most k elements of O for which

$$\{u_{i_1}, \ldots, u_{i_r}\} \preceq \{u_{j_1}, \ldots, u_{j_s}\} \text{ if and only if } \sum_{t=1}^{r} w(u_{i_t}) \leq \sum_{t=1}^{s} w(u_{i_t}).$$

Such an ordering exists; moreover, it can be determined by some rules.

For every subset $\{u_{i_1}, \ldots, u_{i_r}\} \subseteq O$, let

$$O_{\{u_{i_1},\ldots,u_{i_r}\}} = \{u_{i_1}, \ldots, u_{i_r}\} \cup \{u_t : u_t \in O \;\&\; w(u_t) \leq w(u_{i_1})\},$$

where it is supposed that u_{i_1} has the smallest index in $\{u_{i_1}, \ldots, u_{i_r}\}$. Furthermore, let $\mathbf{M}_{\{u_{i_1},\ldots,u_{i_r}\}} = (P, R, O_{\{u_{i_1},\ldots,u_{i_r}\}})$. Then, the following assertion is valid.

Lemma 6.2 *If for some $\{u_{i_1}, \ldots, u_{i_r}\} \subseteq O$, $S(\mathbf{M}_{\{u_{i_1},\ldots,u_{i_r}\}})$ has the maximal structure and for every subset $\{u_{j_1}, \ldots, u_{j_s}\} \subseteq O$ with $\{u_{j_1}, \ldots, u_{j_s}\} \preceq \{u_{i_1}, \ldots u_{i_r}\}$, $S(\mathbf{M}_{\{u_{j_1},\ldots,u_{j_s}\}})$ has no maximal structure, then the maximal structure of $S(\mathbf{M}_{\{u_{i_1},\ldots,u_{i_r}\}})$ is an optimal solution of (4), and the optimal value is $\sum_{t=1}^{r} w(u_{i_t})$.*

On the basis of Lemma 6.2, one can determine an optimal solution of (4) by the following procedure.

Procedure 2.

- *Step 1.* Establish the corresponding linear ordering.
- *Step 2.* Let $i = 1$.
- *Step 3.* Consider the i-th subset of O regarding the fixed ordering. Let $\{u_{i_1}, \ldots, u_{i_r}\}$ be this subset. Perform the Algorithm for Maximal Structure Generation for $\mathbf{M}_{\{u_{i_1},\ldots,u_{i_r}\}}$. If the maximal structure was found, then terminate; the maximal structure is an optimal solution. In the opposite case, let $i := i + 1$ and repeat Step 3.

The time complexity of this procedure is $\sum_{t=1}^{k} \binom{n}{t} \cdot q$ where $q(n)$ denotes the time complexity of the Algorithm for Maximal Structure Generation.

It is worth noting that the technique presented in this section is suitable to solve a generalized version of (4). Namely, if the object function is not the sum of the weights of the k weightest operating units but it only depends on them, *i.e.*, it has a form $z(w(u_{i_1}), \ldots, w(u_{i_k}))$, where the fictious operating units are allowed, then we obtain the following problem:

$$\min\{z(w(u_{i_1}), \ldots, w(u_{i_k})) : (m, o) \in S(\mathbf{M})\}. \tag{5}$$

In this case, the linear ordering has to satisfy the following condition:

$$\{u_{i_1}, \ldots, u_{i_r}\} \preceq \{u_{j_1}, \ldots, u_{j_s}\} \text{ iff}$$

$$z(w(u_{i_1}), \ldots, w(u_{i_k})) \leq z(w(u_{j_1}), \ldots, w(u_{j_k})).$$

Although the bottleneck PNS problem is well-solvable, the minsum PNS problem is NP – complete. Our method for solving of k-sum version of PNS problem is polynomial for "small" fixed k.

7 Application of a Blossom-type Algorithm for the Minimum Cost Edge Covering problem

In some cases we can exploit the fact that the set of input materials and the set of output materials of the operating units do not contain too many elements. Our goal in [9] and [10] was to present heuristic solution methods to PNS problems using only simplified operating units.

7.1 Substitution of operating units with simpler ones

We say that an operating unit is *simple* if the number of its input and output materials is at most 3. For an arbitrary P-graph, we construct an equivalent P-graph consisting of simple operating units. This problem is called the *simplified* PNS *problem.* To construct the simplified problem some new operating units and materials need to be introduced, but the increase in size is not drastic. Let $\mathbf{M} = (P, R, O)$ be an arbitrary PNS problem and let $u = (\alpha, \beta) \in O$ be an operating unit with the sets $\alpha = \{X_1, \ldots, X_n\}$ and $\beta = \{Y_1, \ldots, Y_m\}$. If an operating unit u is not simple, i.e., $(n = 2, m = 2)$, $(n > 2)$ or $(m > 2)$, then we define the new operating units $u_1, \dots, u_{n+m}$ as

$$
\begin{aligned}
&u_1 = (\{X_1, X_2\}, \{Z_1\}), && \\
&u_i = (\{X_{i+1}, Z_{i-1}\}, \{Z_i\}) && i = 2, 3, ..., n-1, \\
&u_n = (\{Z_{n-1}, Z_{n+m}\}, \{Z_n\}), && \\
&u_{n+j} = (\{Z_{n-1+j}\}, \{Z_{n+j}, Y_j\}) && j = 1, 2, ..., m,
\end{aligned}
$$

where $Z_1, Z_2, \dots, Z_{n+m}$ denote new materials and $Z_1, Z_2, \dots, Z_{n+m} \notin M \cup O$. Next, we replace the operating unit u with the operating units $u_1, u_2, \dots, u_{n+m}$ in the set O and let $O^* = O \backslash \{u\} \cup \{u_1, u_2, \dots, u_{n+m}\}$ be the new set of operating units and $\{Z_1, Z_2, \dots, Z_{n+m}\}$ be the new materials which are added to the set M. By making the substitutions of all non-simple units with simple units, we get new disjoint operating unit sets $\{u_1, u_2, \dots, u_{n+m}\}$ and disjoint new material sets $\{Z_1, Z_2, \dots, Z_{n+m}\}$ for any two different units. Let $\mathbf{M}^* = (P^*, R^*, O^*)$ be the derived new PNS problem where the set of product $P^* = P$ and the set of raw materials $R^* = R$.

Theorem 7.1 *There exists a bijective mapping of $S(\mathbf{M})$ onto $S(\mathbf{M}^*)$.*

7.2 Application of the Algorithm for Minimum-cost Edge Covering

Some heuristic algorithms are developed for the simplified PNS problem. In every iteration step the procedure has a set of materials called the set of *required materials*. (In the first step this set is equal to P.) The procedure constructs a graph G for this set in the following way. Two required materials are connected with an edge if there exists an operating unit, the so-called *parent operating unit*, which has these two materials as outputs.

Recall that we have several operating units with exactly two materials as outputs. The cost of this edge is the weight of the minimal weight parent operating unit. Excluding the isolated vertices and applying this algorithm, an edge cover is counted. Then the succeeding set of the required materials contains the materials which are input materials for the current collection of operating units or materials excluded as singular points in this step, but they are not output materials for any operating unit which has already been selected in an iteration step. The procedure is terminated when the current set of required materials is empty. Then the operating units which have been selected in some iteration step constitute a feasible solution.

What is the idea behind of our heuristic algorithm? There is no way of finding an efficient algorithm for the solution of a PNS problem because it is an NP-hard (see [2]) problem. In some well-selected particular cases we can find useful results [3], [4], [7].

In general, heuristics seem to be suitable tools. In what follows we search for an optimal set of operating units which produce the required material set at every step. But we prefer an operating unit if its two outputs are the required material set. Our main tool for the exact optimization at every step is a Blossom-type algorithm for the Minimum Cost Edge Covering (MCEC) (see [34] and [35]). This is fairly similar to the famous Edmonds' matching algorithm. But for this algorithm we can find very few applications in graph theory.

7.3 Heuristics for simplified PNS problems

Let (P, R, O) be the structural model of a simplified-type PNS problem. R: raw materials, P: desired products, $P \cap R = \emptyset$, $M \supseteq P \cup R$ the set of materials. The type of every operating unit $u \in O$ is $(1, 2)$ or $(2, 1)$. For every $m_q \in M \setminus R$ there exists $(\alpha, \beta) \in O$ such that $m_q \in \beta$.

Algorithm 1:

Notation:

RM_i : Required Materials after Step i.
PM_i : Produced Materials after Step i.
$G_i(N_i, A_i)$ $(c(e) : e \in A_i)$.
We connect two material points $m_q, m_r \in RM_{i-1}$, with an edge e iff there exist $(\alpha, \beta) \in O$ and $m_q, m_r \in \beta$ $(\beta = \{m_q, m_r\})$.
$(\alpha, \beta)_{q,r} :=$ one of the minimal weight operating units which produces m_q and m_r, such an operating unit exists.
$c(e) = w(\alpha, \beta)_{q,r}$ if $e = (m_q, m_r)$.
$A_i = \{e : e = (m_q, m_r)\ m_q, m_r \in RM_{i-1}\ m_q \neq m_r$
and $\exists(\alpha; m_q, m_r) \in O\}$.
If $A_i = \emptyset$ then G_i doesn't exist.
N_i ={nodes incident with edges of A_i} $(\subseteq RM_{i-1})$.

Step 1: *Initialization* $O_0 := \emptyset, RM_0 := P$,
$PM_0 := R, i := 0$

Step 2: *Construction of* G_i

If there exists a G_i, (apply Algorithm MCEC to G_i), $\longrightarrow$ Step 3.

If G_i doesn't exist $\longrightarrow$ Step 4

Step 3: Apply Algorithm MCEC to G_i, and let E_i denote the solution set of edges. Let

$O_i := O_{i-1} \cup \{(\alpha, \beta)_{q,r} : (m_q, m_{r)} \in E_i\}$,

$RM_i := (P \cup mat^{in}O_i) \setminus mat^{out}O_i$,

$PM_i := R \cup mat^{out}O_i$.

If $RM_i = \emptyset$, then $O^1 = O_i$, the solution set of operating units of Algorithm 1.

If $RM_i \neq \emptyset$, then $i := i + 1 \longrightarrow$ Step 2.

Step 4: Let u^i one of minimal weight operating units which produce one of materials of RM_{i-1}. Moreover, let

$O_i := O_{i-1} \cup u^i$,

$RM_i := (P \cup mat^{in}O_i) \setminus mat^{out}O_i$,

$PM_i := R \cup mat^{out}O_i$,

If $RM_i = \emptyset$, then $O^1 = O_i$, the solution set of operating units of Algorithm 1.

If $RM_i \neq \emptyset$, then $i := i + 1 \longrightarrow$ Step2.

In [9] and [10] we defined more modified algorithms.

8 Domination sets in de Bruijn graphs

The concept of a perfect d-dominating set have appeared first in a paper [1] by Biggs, who introduced the term perfect d-code to denote what we call a perfect d-dominating set. He defined the distance-transitive graphs and derived an important necessary condition for the existence of a perfect d-code in such graphs. In [31] M. Livingston and Q. F. Stout have studied other families of graphs arising from the interconnection networks of parallel computers. They proved a theorem about the existence of a dominating set of some de Bruijn graphs, but leaved open questions on other infinite de Bruijn graph classes. We proved in [11] two conjectures. In this section we study different type of dominating sets in de Bruijn graphs and we prove conjectures on perfect dominating sets.

8.1 Notions

Let $|\mathcal{A}| = n$. The de Bruijn graph is defined as:

$$B(n,k) = (V(n,k), E(n,k))$$

with $V(n,k) = \mathcal{A}^k$ as the set of vertices, and $E(n,k) = \mathcal{A}^{k+1}$ as the set of directed arcs. There is an arc from $x_1x_2\dots x_k$ to $y_1y_2\dots y_k$ if $x_2x_3\dots x_k = y_1y_2\cdots y_{k-1}$.

In a graph $G = (V,E)$ a vertex y is *dominated* by a vertex x (or x *dominates* y) if there exists an arc from x to y or $x = y$. A set of vertices $D \subseteq V$ is a *dominating set* of G if every vertex of G is dominated by at least one vertex of D. The size of a set of least cardinality among all dominating sets for G is called the *domination number* of G and any dominating set of this cardinality is called a *minimum dominating set* for G. When each vertex of G is dominated by exactly one element of D then the set D is called a *perfect dominating set* of G. A vertex x *d-dominate* a vertex y if there is a path from x to y in G of length at most d. A set D of vertices is a *d-dominating set* in G if each vertex of G is d-dominated by at least one vertex of D. This set D is a *perfect d-dominating set (d-PDS)* if each vertex of G is d-dominated by exactly one vertex of D.

8.2 Minimum dominating sets

In general graphs every domination problem is NP – complete. We have a constructive result about minimum dominating sets in directed de Bruijn graphs.

Theorem 8.1 [11] *In the de Bruijn graph $B(2,k)$ a minimum dominating set has $\left\lceil \frac{2^k}{3} \right\rceil$ vertices.*

Theorem 8.2 [11] *In the de Bruijn graph $B(n,k)$ a minimum dominating set has $\left\lceil \frac{n^k}{n+1} \right\rceil$ vertices.*

M. Livingston and Q. F. Stout proved in [31] the following result (Theorem 2.12).

Statement 8.3 *For any $d \geq 1$ and for k a positive integer of the form $(d+1)m$ or $(d+1)m-1$ or $k < d$, let T_k denote a subset of the vertices of $B(2,k)$ defined as*

(i) $T_1 = T_2 = \ldots = T_d = \{0\}$,
(ii) $T_{(d+1)(m+1)-1} = T_{(d+1)m-1} \cup \{j : 2^{(d+1)m-1} \leq j \leq 2^{(d+1)m} - 1\}$,
(iii) $T_{(d+1)m} = T_{(d+1)m-1} \cup \{2^{(d+1))m} - 1 - s : s \in T_{(d+1)m-1}\}$.

Then the set T_k is a perfect d-dominating set for $B(2,k)$.

In [31] the following conjecture was set out: *there is no perfect 2-dominating set for $B(2,k)$, when $(k-1)$ is a multiple of 3.* We proved this conjecture in [11]:

Theorem 8.4 [11] *In the de Bruijn graph $B(2,k)$ there is no perfect 2-dominating set if $(k-1)$ is a multiple of 3.*

M. Livingston and Q. F. Stout in [31] considered the undirected case, too. We can define the undirected version of the de Bruijn graph $B(n,k)$ if we change arcs to undirected edges. Let us denote by $B^*(n,k)$ the undirected de Bruijn graph. Now there is an edge between $x_1x_2\ldots x_k$ and $y_1y_2\ldots y_k$ if $x_2x_3\ldots x_k = y_1y_2\ldots y_{k-1}$ or $x_1x_2\ldots x_{k-1} = y_2y_3\ldots y_k$. We can give all definitions about *domination* (and *d-domination*) in a very similar way. In [31] we can read the fact that the undirected de Bruijn graph $B^*(2,k)$ has a perfect dominating set (PDS) for k=1 or 2, but has no PDS for k=3, 4, or 5. In [11] we proved that $B^*(2,k)$ has PDS only for k=1 or 2.

Theorem 8.5 [11] *There is no PDS in $B^*(2,k)$ if $k > 2$.*

9 The HPPIT problem

In many important combinatorial optimization problems it is required to find a permutation of vertices of a complete directed graph that minimizes a certain cost function. The most familiar one is the min-cost Hamiltonian path problem, or its closed-path version, the Traveling Salesman Problem (TSP). Another problem known as the Linear Ordering Problem (LOP) is to find a linear order of the nodes of a directed graph such that the sum of the arc weights, which are consistent with this order, is as large as possible.

9.1 Common generalization of TSP and LOP problems

In [12] we considered a new optimization model using a mixed linear cost function from these two. The motivation of this common generalization of TSP and LOP is the following practical question. We consider the problem where a vehicle has to visit some places but it can be used for internal transports during its tour. The goal is to find an ordering of the places which maximize the total profit which can be achieved by the internal transports. This vehicle routing question leads to the LOP model.

On the other hand we also have to take into account the cost of the tour. We reduce the profit by this cost and the difference gives the objective function of the mathematical model. We call this problem min-cost Hamiltonian path problem with internal transport, $HPPIT$ in short. HPPIT is a generalization of two NP-hard problems therefore it is also NP-hard. For NP-hard optimization problems, the construction and analysis of heuristic algorithms is a rapidly developing area. We extend some heuristic algorithms which are defined for the TSP problem to this more general model, and we develop some further algorithms. The algorithms are analyzed by an empirical analysis [12].

9.2 Notions and notation

Let G(V,A) be a directed complete graph, where $V = v_0, v_1, \dots, v_n$ is the set of vertices, (v_0 is the depot, and the other vertices are the places which should be visited by the vehicle). Furthermore two $(n+1) \times (n+1)$ nonnegative matrices are given, B and D. B_{ij} is the possible profit which can be achieved by the inner transportation from v_i to v_j if v_i is visited before v_j ($B_{ii} = 0$ for each i). D_{ij} gives the cost of travelling from v_i to v_j ($D_{ii} = 0$ for each i).

In the HPPIT problem we would like to find a tour which visits each city exactly once and starts at the depot and returns there at the end of the tour. The objective is to maximize the total profit achieved by the inner transportation taking into account the cost of the tour. A feasible solution can be defined as a permutation p of the set $\{1, .., n\}$. The permutation describes the tour where the vehicle starts and ends at v_0 and visits the other vertices in the order $v_{p(1)}, v_{p(2)}, .., v_{p(n)}$. Then the objective function is given by the formula $z(p) =$

$$\sum_{0<i<j<n+1} B_{p(i),p(j)} + \sum_{i=1}^{n} (B_{0,p(i)} + B_{p(i),0}) - \sum_{0 \le i < n} D_{p(i),p(i+1)} - D_{p(n),0},$$

and the goal is to maximize this function. We can also represent the solutions as the directed cycles of the graph V which contain all of the vertices.

9.3 Tour building algorithms

In [12] we present some heuristic algorithms for the solution of the problem. First we give six tour building algorithms which use different heuristic rules to build a feasible solution. Then a tour improvement algorithm is presented which is based on the neighborhood search technique. In one of these greedy algorithms builds the order in two directions: forward and backward. In each step we choose a vertex backward and one forward to extend the current partial order. We always choose the vertices which yield the maximal profit (taking the travelling cost into account). The algorithm can be defined as follows:

Algorithm 3:

Step 1: (Definition of the last and the first vertices $v_{p(n)}$ and $v_{p(1)}$): Let $0 < k < n+1$ be the value, where $\sum_{0<j<n+1} B_{jk} - D_{k0} = \max_{0<i<n+1} \sum_{0<j<n+1} B_{ji} - D_{i0}$. If more than one k exist with this property, then we choose the largest one. Let $p(n) = k$, $F = \{k\}$ (F is the set of the ordered vertices). If $n > 1$, then let $0 < k < n+1$, $k \notin F$ be the value, where $\sum_{0<j<n+1, j \neq p(n))} B_{kj} - D_{0k} = \max_{0<i<n+1, i \notin F} \sum_{0<j<n+1, j \neq p(n)} B_{ij} - D_{0i}$. Let $p(1) = k$. If more than one k exist with this property, then we choose the largest one. Let $t = n-1$, $z = 2$, and $F = F \cup \{k\}$.

Step 2: We determine p(t), the next element from backward in the tour. If $z = n$, then the procedure is finished, p is defined, and the tour is $v_0, v_{p(1)}, \ldots, v_{p(n)}, v_0$. If $z < n$, then let $p(t)$ be the maximal k, $k \notin F$ such that $-D_{k,p(t+1)} + \sum_{0<j<n+1, j \notin F} B_{jk} = \max_{0<i<n+1, i \notin F}\{-D_{i,p(t+1)} + \sum_{0<j<n+1, j \notin F} B_{ji}\}$. Let $z = z+1$, $t = n-t+1$, $F = F \cup \{k\}$.

Step 3: Determine p(t), the next element forward in the tour. If $z = n$, then the procedure is finished, p is defined, the tour is $v_0, v_{p(1)}, \ldots, v_{p(n)}, v_0$. If $z < n$, then let $p(t)$ be the minimal k, $k \notin F$, such that $-D_{p(t-1),k} + \sum_{0<j<n+1, j \notin F} B_{kj} = \max_{0<i<n+1, i \notin F}\{-D_{p(t-1),i} + \sum_{0<j<n+1, j \notin F} B_{ij}\}$ Let $z = z+1$, $t = n-t$, $F = F \cup \{k\}$, and go to Step 2.

10 Conclusion

In Process Network Synthesis a desired material set P is given, and our goal is to find an operating unit set to produce P. For every unit we have a material set as input set of the unit, which are need to be a subset of the dominated set of all units union raw material set. In a manufacturing system or in other practical application of the PNS model we need not a unit if it is not d-dominate some material from P, for a positive integer d. The role of dominating sets in topic of process networks presents in the original sense only in chapter 8. The third topic of the thesis is the HPPIT problem. One of the two parent problems is the LOP. We need to count all the arcs forward in the case of constant weight function. The question is, how many the sum of the number of dominated vertices consistent to the order, and which ordering give a maximal value.

The second chapter is a survey of basic notions and notations of PNS. The definition of the structural model, the first combinatorial properties of the feasible solution processes, the notion of maximal structure were in [14], [15], [16], [17], [18], [19], [21], [24], [25] papers. The first goal was to find an appropriate feasible solution with minimal sum of costs of units in it. This objective function yields the so-called minsum version of the weighted PNS problems. For the solution of this PNS problem, more algorithms were developed, most of them are based on Branch and Bound technique.

How can we solve a PNS problem efficiently? Can we use combinatorial ideas for an algorithm to solve PNS in polynomial time? In chapter 3 we were able to show a nice polynomial transformation of a special case of Minsum PNS problem (2) to the set covering problem [2]. We proved this fundamental question of theory of PNS, the problem is NP–hard (Theorem 3.2)!

Exponential time Branch and Bound algorithms were studied for PNS problem in [15]. In chapter 4 we considered the bounding problem of the number of consistent decision-mappings belonging to an $\mathbf{M} = (P, R, O)$ structural model. It is important to improve bounding function, and to put smaller the space of maximal consistent decision-mappings of a Branch and Bound technique. Taking into account axiom ($\mathcal{A}2$) only, this bound can be improved (Theorem 4.4). In a special PNS class we have given a complicated formula for $|A_I|$, see [3]. We considered two more special models, the so-called Line model and Chain model. We could count a formula for these bounds with Inclusion–Exclusion Formula and with direct way, too. So we obtained two nice combinatorial identity from these counting.

The general Minmax or Bottleneck optimization problem and the k-sum versions are NP–complete problems. The Minsum version of the PNS problem is NP–complete, what we can tell about complexity of these two versions of PNS? In [6] we answered these questions. We have Procedure 1. based on Lemma (6.1) which solve the Bottleneck optimization problem for PNS efficiently. For the k-sum version of PNS we proved Lemma 6.2 and gave an algorithm to solve this problem, too. Our method for solving of k-sum version of PNS problem is polynomial for “small” fixed k.

Every P-graph can be transformed into a simplified form. This simplified form consist of simple operating units in which the total number of input and output materials is at most 3. This observation facilitates useful application of the Edge Covering Problem of weighted graphs, which can be used to develop a new heuristic procedure for the PNS problem. In chapter 7 we prove Theorem 7.1 about an equivalent transformation of P-graph. Using the algorithm MCEC we have defined 4 algorithms for simplified PNS problems, and gave an analysis of our computational experiments see [9] and [10]. Our main tool for the exact optimization at every step of heuristics was the algorithm MCEC, which is a blossom-type algorithm, it is very similar to the famous Edmonds' matching algorithm.

In [31] M. Livingston and Q. F. Stout gave a construction of a PDS for de Bruijn graphs in infinitely many cases, but their characterization was not complete. We have proved two theorems, 8.4 and 8.5 about their conjectures. These results claim for infinite k parameter values that some directed and undirected de Bruijn graphs with parameter k if there have a 2-PDS or PDS, [11]. We have a construction for a minimal dominating set in directed de Bruijn graphs in general.

In chapter 9 we consider a new combinatorial optimization model as a common generalization of TSP and LOP problems. These two problem are well-known NP–complete problems. Why I define HPPIT, a more complex problem with a mixed linear cost function from the parent problems? The motivation was given by practical optimization questions. The objective was to maximize the total profit achieved by the inner transportation taking into account the cost of the tour of a vehicle.

Thesis 1

Theorem 3.1 [2] The class PNS_{w1} is equivalent to the class of the set covering problems.

Corollary 3.2 [2] The PNS problem is NP-hard.

Thesis 2

Theorem 4.2 [3] For every $\emptyset \neq m \subseteq M \setminus R$, $\tau(m) = 2^{|\cup\{\Delta(X):X\in m\}|}$.

Theorem 4.3 [3] $\tau'(m) = |\Omega_{\mathbf{M}}^{\max} \setminus (A_1 \cup A_2 \cup ... \cup A_k)| =$
$= \Sigma_{I\subseteq\{1,...k\}}(-1)^{|I|} \cdot |A_I|$.

Theorem 5.1 [3] For separator type operating units

$$|A_I| = \left(\prod_{t=1}^{l}\left(2^{|O^*(X_{i_t})|} - 1\right)\right) \cdot 2^{|O\setminus(\cup_{i\in I}\Delta(X_i))\setminus(\cup_{i\in I}O(X_i))|}.$$

Theorem 5.2 [4] In the Line model $|S(\mathbf{M})| \leq L^{(1)}$

Theorem 5.3 [4] In the Chain model $|S(\mathbf{M})| \leq C^{(1)}$

Thesis 3

Lemma 6.1 [6] If for some integer $1 \leq i \leq n$, $S(\mathbf{M}_i)$ has the maximal structure and $S(\mathbf{M}_{i-1})$ does not have the maximal structure, then u_i is included in every feasible solution in $S(\mathbf{M}_i)$.

Procedure 1. [6]

Lemma 6.2 [6] If for some $\{u_{i_1}, \ldots, u_{i_r}\} \subseteq O$, $S(\mathbf{M}_{\{u_{i_1},\ldots,u_{i_r}\}})$ has the maximal structure and for every subset $\{u_{j_1}, \ldots, u_{j_s}\} \subseteq O$ with $\{u_{j_1}, \ldots, u_{j_s}\} \preceq \{u_{i_1}, \ldots u_{i_r}\}$, $S(\mathbf{M}_{\{u_{j_1},\ldots,u_{j_s}\}})$ has no maximal structure, then the maximal structure of $S(\mathbf{M}_{\{u_{i_1},\ldots,u_{i_r}\}})$ is an optimal solution of (4), and the optimal value is $\sum_{t=1}^{r} w(u_{i_t})$.

Procedure 2. [6]

Thesis 4

Theorem 7.1 [9] There exists a bijective mapping of $S(\mathbf{M})$ *onto* $S(\mathbf{M}^*)$.

Algorithm 1–4

Thesis 5

Theorem 8.1 [11] In the de Bruijn graph $B(2,k)$ a minimum dominating set has $\left\lceil \frac{2^k}{3} \right\rceil$ vertices.

Theorem 8.2 [11] In the de Bruijn graph $B(n,k)$ a minimum dominating set has $\left\lceil \frac{n^k}{n+1} \right\rceil$ vertices.

Theorem 8.4 [11] In the de Bruijn graph $B(2,k)$ there is no perfect 2-dominating set if $k-1$ is a multiple of 3.

Theorem 8.5 [11] There is no PDS in $B^*(2,k)$ if $k > 2$.

Thesis 6

Definition of the HPPIT problem. [12]

Algorithm 1-4

Publications of the results

Blázsik, Z., B. Imreh, A note on connection between PNS and set covering problems, *Acta Cybernetica*, **12**, 1996, 309-312. (MR1428741)

Blázsik, Z., Cs. Holló, B. Imreh, On Decision-Mappings Related to Process Network Synthesis Problem, *Acta Cybernetica*, **13**, 1998, 319-328. (MR1644388)

Blázsik, Z., Cs. Holló, B. Imreh, Explicit bound for the number of feasible solutions of special PNS-problem classes, *Pure Mathematics and Applications*, **9**, 1998, 17-27. (MR1677229)

Blázsik, Z., Cs. Holló, B. Imreh, Cs. Imreh, Z. Kovács, On Bottleneck and k-sum version of the Process Network Synthesis Problem, *Novi Sad Journal of Mathematics*, **3**, 2000, 11-19. (MR1776440)

Blázsik, Z., K. Keserű, Z. Kovács, Heuristics for simplified Process Network Synthesis problems with a Blossom-type Algorithm for the edge covering problem, Optimization Theory: Recent Developments from Matrahaza,(eds.: F. Gianessi, P. Pardalos, T. Rapcsák), Kluwer Academic Publishers, Dordrecht, 2001, 19-31. (MR1886425)

Blázsik, Z., K. Keserű, Z. Kovács, Heuristics for PNS problems and its empirical analysis, *Pure Mathematics and Applications*, **11**, 2001, 139-151. (MR1839923)

Blázsik, Z., Z. Kása, Dominating sets in de Bruijn graphs. Algebraic systems (Felix-Oradea, 2001). *Pure Mathematics and Applications*, **13**, 2002, 79-85. (MR1987200)

Blázsik Z., T. Bartók, B. Imreh, Cs. Imreh, Z. Kovács, Heuristics on a Common Generalization of TSP and LOP, accepted for publication.

References

[1] N. Biggs, Perfect codes in graphs, *J. Comb. Theory* (B), **15**, 1973, 289-296.

[2] Blázsik, Z., B. Imreh, A note on connection between PNS and set covering problems, *Acta Cybernetica*, **12**, 1996, 309-312.

[3] Blázsik, Z., Cs. Holló, B. Imreh, On Decision-Mappings Related to Process Network Synthesis Problem, *Acta Cybernetica*, **13**, 1998, 319-328.

[4] Blázsik, Z., Cs. Holló, B. Imreh, Explicit bound for the number of feasible solutions of special PNS-problem classes, *Pure Mathematics and Applications*, **9**, 1998, 17-27.

[5] Blázsik, Z., Cs. Holló, B. Imreh, Kiszámolható korlátok speciális PNS-problémaosztályok lehetséges megoldásai számára, *Új utak a magyar operációkutatásban, szerk. Komlósi, S., Szántai T., Dialóg Campus Kiadó, Budapest-Pécs,* 1999, 182-194.

[6] Blázsik, Z., Cs. Holló, B. Imreh, Cs. Imreh, Z. Kovács, On Bottleneck and k-sum version of the Process Network Synthesis Problem, *Novi Sad Journal of Mathematics*, **3**, 2000, 11-19.

[7] Blázsik, Z., Cs. Holló, B. Imreh, Cs. Imreh, Z. Kovács, On a well-solvable class of the PNS problem, *Novi Sad Journal of Mathematics*, **3**, 2000, 21-30.

[8] Blázsik, Z., Cs. Holló, Cs. Imreh, Z. Kovács, Heuristics for the Process Network Synthesis Problem, *New Trends in Equilibrium Systems, Mátraháza Optimization Days*, Kluwer Academic Publishers, 2000, 1-16.

[9] Blázsik, Z., K. Keserű, and Z. Kovács, Heuristics for simplified Process Network Synthesis problems with a Blossom-type Algorithm for the edge covering problem, Optimization Theory: Recent Developments from Matrahaza,(eds.: F. Gianessi, P. Pardalos, T. Rapcsák), Kluwer Academic Publishers, Dordrecht, 2001, 19-31.

[10] Blázsik, Z., K.. Keserű, and Z. Kovács, Heuristics for PNS problems and its empirical analysis, *Pure Mathematics and Applications*, **11**, 2001, 139-151.

[11] Blázsik, Z., Z. Kása, Dominating sets in de Bruijn graphs. Algebraic systems (Felix-Oradea, 2001). *Pure Mathematics and Applications*, **13**, 2002, 79-85.

[12] Blázsik Z., T. Bartók, B. Imreh, Cs. Imreh, Z. Kovács, Heuristics on a Common Generalization of TSP and LOP, accepted for publication.

[13] Floudas, C. A., I. E. Grossmann, Algorithmic Approaches to Process Synthesis: Logic and Global Optimization, AIChE Symposium Series No. 304, **91** (Eds: L. T. Biegler and M. F. Doherly), 1995, 198-221.

[14] Friedler, F., K. Tarján, Y. W. Huang, and L. T. Fan, Graph-Theoretic Approach to Process Synthesis: Polynomial Algorithm for maximal structure generation, *Computer chem. Engng.* **17**, 1993, 924-942.

[15] Friedler, F., J. B. Varga, and L. T. Fan, Decision-Mappings: A Tool for Consistent and Complete Decisions in Process Synthesis, *Chem. Eng. Sci.*, **50** (11), 1995, 1755-1768.

[16] Friedler, F., J.B. Varga, E. Fehér, and L.T. Fan, Combinatorially Accerelated Branch-and-Bound Method for Solving the MIP Model of Process Network Synthesis, International Conference on State of the Art in Global Optimization: Computational Methods and Applications, Princeton, 1995.

[17] Friedler, F., J. B. Varga, E. Fehér, L. T. Fan, Combinatorially Accelerated Branch-and -Bound Method for Solving the MIP Model of Process Network Synthesis, *Nonconvex Optimization and its Applications*, (eds.: C. A. Floudas and P. M. Pardalos), Kluwer Academic Publishers, Norwell, MA, U.S.A., 1996, 609-626.

[18] Friedler, F., K. Tarján, Y. W. Huang, L. T. Fan, Graph-Theoretic Approach to Process Synthesis: Axioms and Theorems, *Chem. Eng. Sci.*, **47** (8), 1992, 1973-1988.

[19] Friedler, F., K. Tarján, Y. W. Huang, L. T. Fan, Combinatorial Algorithms for Process Synthesis, *Computer chem. Engng.*, **16**, 1992, 313-320.

[20] Friedler, F., J. Fülöp, B. Imreh, On the reformulation of some classes of PNS-problems as set covering problems, *Acta Cybernetica*, **13**, 1998, 329-337.

[21] Friedler, F., L. T. Fan, B. Imreh, Process Network Synthesis: Problem Definition, *Networks*, **28**, 1998, 119-124.

[22] Grossmann, I. E., V. T. Voudouris, O. Ghattas, Mixed-Integer Linear Programming Reformulations for Some Nonlinear Discrete Design Optimization Problems, In: Recent Advances in Global Optimization (Eds: C. A. Floudas and P. M. Pardalos) Princeton University Press, New Jersey, 1992.

[23] Holló, Cs., Z. Blázsik, Cs. Imreh, Z. Kovács, On a Merging Reduction of the Process Network Synthesis Problem, *Acta Cybernetica*, **14**, 1999, 251-261.

[24] Imreh, B., F. Friedler, L. T. Fan, An Algorithm for Improving the Bounding Procedure in Solving Process Network Synthesis by a Branch-and-Bound Method, *Developments in Global Optimization*, ed. I. M. Bomze, T. Csendes, R. Horst, P. M. Pardalos, Kluwer Academic Publisher, Dordrecht, Boston, London, 1996, 301-348.

[25] Imreh, B., G. Magyar, Empirical Analysis of Some Procedures for Solving Process Network Synthesis Problem, *Journal of Computing and Information Technology*, **6**, 1998, 373-382.

[26] Imreh, B., *Kombinatorikus optimalizálás*, Novadat, Győr, 2000.

[27] Imreh, B., J. Fülöp, F. Friedler, A note on the Equivalence of the Process Network Synthesis and Set Covering problems, *Acta Cybernetica*, **14**, 2000, 497-502.

[28] Imreh, Cs., Jól megoldható PNS osztályokról, *Új utak a magyar operációkutatásban, szerk. Komlósi, S., Szántai T., Dialóg Campus Kiadó, Budapest-Pécs*, 1999, 168-181.

[29] Imreh, Cs., A new well-solvable class of PNS problems, *Computing*, **66**, 2001, 289-296.

[30] Karp R. M., Reducibility among Combinatorial Problems in Complexity of Computer Computations, *R. E. Miller and T. W. Thatcher, eds.*, Plenum Press, New York, 1972.

[31] Livingston, M., Q. F. Stout, Perfect dominating sets, *Congr. Numer.*, **78**, 1990, 187-203.

[32] Lothaire, M., *Combinatorics on words*, Addison-Wesley, Reading, 1983.

[33] de Luca, A., On the combinatorics of finite words, *Theor. Comput. Sci.*, **218**, 1999, 13-39.

[34] Murty, Katta G., Clovis Perin, A 1-Matching Blossom-Type Algorithm for Edge Covering Problems, *Networks*, **12**, 1982, 379-391.

[35] Murty, Katta G., Network Programming, *Prentice Hall*, 1992.